CLAIMS:

What is claimed is:

1. A method for managing direct memory access resources, the method comprising:

responsive to a change in context for a direct memory access resource, storing data relating to the switch in a context switch history containing a number of prior context switches occurring prior to a current context; and

freeing portions of the direct memory access chain of requests using the context switch history to form freed portions, wherein the freed portions are reused for requests.

2. The method of claim 1, wherein the portions are direct memory access queues.

3. The method of claim 2, wherein the freeing step comprises:

responsive to processing of requests, identifying a direct memory access queue for contexts prior to the current context to form an identified direct memory access queue; and

freeing the identified direct memory access queue.

4. The method of claim 1, wherein the context switch history includes a pointer to a private structure for rendering the context, a pointer to a registration

structure for pools of queues linked to the context, and an identifier for the thread.

5. The method of claim 1, wherein the context switch history is a circular list of a number of context switches.

6. The method of claim 1, wherein the change in context occurs when the direct access memory resource is available by a first thread and access to the direct access memory resource is granted to a second thread and wherein the direct access memory resource is made available to the second thread by adding a buffer of the second thread to an end of a direct memory access chain of requests for the first thread to generate a direct memory access request for the second thread.

7. The method of claim 6, wherein the direct memory access request is a zero length direct memory access request.

8. The method of claim 2 further comprising:

responsive to encountering an error from a parameter in a request in the direct memory access chain of requests, identifying a queue originating the parameter in the direct memory access chain of requests using the context switch history, wherein a bad hardware address is identified using the queue.

9. A data processing system for managing direct memory access resources, the data processing system comprising:

storing means, responsive to a change in context for a direct memory access resource, for storing data relating to the switch in a context switch history containing a number of prior context switches occurring prior to a current context; and

freeing means for freeing portions of the direct memory access chain of requests using the context switch history to form freed portions, wherein the freed portions are reused for requests.

10. The data processing system of claim 9, wherein the portions are direct memory access queues.

11. The data processing system of claim 10, wherein the freeing means is a first freeing means and further comprising:

identifying means, responsive to processing of requests, for identifying a direct memory access queue for contexts prior to the current context to form an identified direct memory access queue; and

second freeing means for freeing the identified direct memory access queue.

12. The data processing system of claim 10 further comprising:

identifying means, responsive to encountering an error from a parameter in a request in the direct memory access chain of requests, for identifying a queue

originating the parameter in the direct memory access chain of requests using the context switch history, wherein a bad hardware address is identified using the queue.

13. A computer program product in a computer readable medium for managing direct memory access resources, the computer program product comprising:

first instructions, responsive to a change in context for a direct memory access resource, for storing data relating to the switch in a context switch history containing a number of prior context switches occurring prior to a current context; and

second instructions for freeing portions of the direct memory access chain of requests using the context switch history to form freed portions, wherein the freed portions are reused for requests.

14. The computer program product of claim 13, wherein the portions are direct memory access queues.

15. The computer program product of claim 14, wherein the second instructions comprises:

first sub-instructions, responsive to processing of requests, for identifying a direct memory access queue for contexts prior to the current context to form an identified direct memory access queue; and

second sub-instructions for freeing the identified direct memory access queue.

16. The computer program product of claim 13, wherein the context switch history includes a pointer to a private structure for rendering the context, a pointer to a registration structure for pools of queues linked to the context, and an identifier for the thread.

17. The computer program product of claim 13, wherein the context switch history is a circular list of a number of context switches.

18. The computer program product of claim 13, wherein the change in context occurs when the direct access memory resource is available by a first thread and access to the direct access memory resource is granted to a second thread and wherein the direct access memory resource is made available to the second thread by adding a buffer of the second thread to an end of a direct memory access chain of requests for the first thread to generate a direct memory access request for the second thread.

19. The computer program product of claim 18, wherein the direct memory access request is a zero length direct memory access request.

20. The computer program product of claim 14 further comprising:

third instructions, responsive to encountering an error from a parameter in a request in the direct memory access chain of requests, for identifying a queue

originating the parameter in the direct memory access chain of requests using the context switch history, wherein a bad hardware address is identified using the queue.